Piano Notes Extraction using FFT Algorithm

Syed Imaduddin
Department of Electronics Engineering
Aligarh Muslim University
Aligarh, India
syedimaduddin@zhcet.ac.in

Ashish Kumar
Department of Electronics Engineering
Aligarh Muslim University
Aligarh, India
ashuvaire@gmail.com

***Abstract*—This research paper focuses on identifying musical notes using digital signal processing techniques. The authors discuss the challenges involved in recognizing and analyzing musical information, particularly for learners and computers. The paper provides an overview of the fundamental concepts of musical notation and discusses various methods proposed by researchers to identify notes in audio files, including event detection and pitch extraction. The study concludes that digital signal processing algorithms can be effective for identifying musical notes, particularly for piano songs, and have significant potential for practical applications in music education and composition.**

***Index Terms*—Piano Notes Extraction, Digital Signal Processing, FFT Algorithm, Spectral Analysis, Frequency Extraction, Averaging Filter, Sampling Theorem**

I. Introduction

The emergence of personal digital technology has revolutionized the distribution and storage of music, leading to an increased interest in and focus on how information technology can be utilized in relation to music content. While trained listeners can easily derive musical information from live or recorded performances, this task poses significant challenges for learners and computers. Therefore, there is a need for a quick, automated, and error-free method to obtain this information for practical applications. Every song has a specific tempo or speed at which it should be played. This tempo is expressed as beats per minute, where a beat is typically defined as a specific duration of a note. The value assigned to each note (such as quarter or half) determines its duration in beats. An eighth note refers to a half beat and a quarter note to one beat. Music notation typically uses seven letters to represent notes: A, B, C, D, E, F, and G. However, each letter can have different frequencies within an octave, which is a range of pitches that can be doubled or halved. To modify a note's pitch, sharps (#) are used to raise it by a half-step, and flats (b) are used to lower it by the same amount. Rests indicate periods of silence within a piece of music, and placing a dot next to a note or rest lengthens its duration by half its original value. The human ear is capable of perceiving signal frequencies that fall within the range of 20 Hz to 20 kHz. Some of these frequencies are specifically associated with pianos, which have different ranges depending on the instrument. A piano note's frequency range is usually a complex web of interconnected frequencies, forming a matrix of related frequencies. The frequency spectrum of a musical composition can be graphically represented, and its most prominent frequencies can be identified by analyzing this spectrum. Vibrating strings and bars produce harmonic overtones. The basic frequencies of a piano's keys can vary between 27.5 Hz and 4186 Hz. As per contemporary tuning theory, note A situated above the middle C should have a frequency of 440 Hz, commonly known as A-440 in the music domain. The musical scale is composed of a series of notes, and these notes are defined by their corresponding frequencies, as illustrated in figure 1.

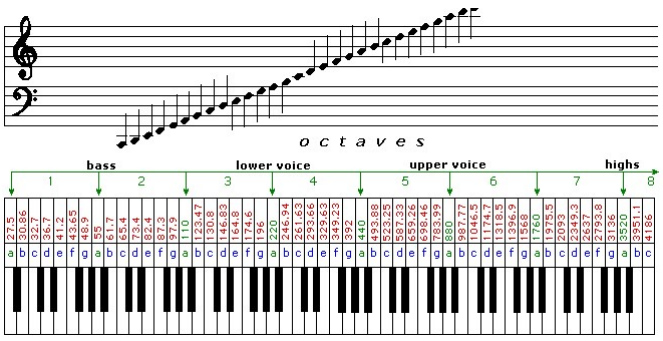

Fig. 1. The frequencies that correspond to the notes in a musical scale [1]

Prasanth L. et. al. [2] have conducted an experiment using their proposed algorithm on piano songs containing unfamiliar notes. This project could be beneficial for individuals who have a passion for music but limited knowledge. The algorithm takes the song as an input, analyzes its features, and utilizes Digital Signal Processing (DSP) to detect and recognize each note, along with its duration. DSP is a technique used to manipulate signals that have been converted from real-world analog signals into digital signals. DSP involves performing mathematical operations such as addition, subtraction, multiplication, and division on these digital signals in order to extract, analyze, or transform the information they contain. This processing is essential to convert signals into a format that can be easily displayed, analyzed or used for other purposes.

The paper "Musical Notes Identification using Digital Signal Processing" [3] uses audio files as input and processes them to extract features for note identification using digital signal processing techniques. The paper explains how the algorithm works and that only piano songs are allowed as input because the notes are already known. Events are detected to determine where notes begin and end, and pitches are extracted

to determine the pitches played within intervals. The authors address the challenge of different playing speeds by carefully processing signals in both the time and frequency domains. The identified notes are compared to the original notes until a high matching rate is detected.

With the Analysis-by-Synthesis technique, Foo et. al. [4] proposed a method for identifying piano notes that is based on the assumption that multiple notes can be played simultaneously, but that only one signal source is available at each time. An analysis is conducted using the Constant-Q Transform in the frequency domain. In order to resolve ambiguities resulting from notes having overlapping harmonics, a Frequency Response Masking filter (FRM) is used. The filters have been carefully selected for their selectivity and bandwidth. The researchers tested their approach on recordings of professional pianists playing on an acoustic piano, and found that the system can accurately detect up to six simultaneous notes.

Rao et. al. [5] created a software program that can take an audio input, such as a song, and analyze it to output musical notes. This is particularly useful for learning, teaching, and composing music. The software has addressed the challenging task of generating sheet music from audible music, which has been a difficult task for both computers and skilled musicians.

A skilled vocalist must not only sing well, but also capture the essence of a song. Unfortunately, many musical compositions are written in block notation, which poses a challenge for students accustomed to using cipher or numbered notation. To address this issue, Gazali et al. [6] developed an application that utilizes the Fast Fourier Transform (FFT) method to convert digital signals to analog signals. This program can play sheet music entered by the user and accurately reproduce the melody based on the user's auditory perception. However, the program has limitations in that it can only process input within the range of C4 to C5 and cannot accept sharps (#) or flats (b) in the input tones.

The Fourier transformation is a mathematical process that converts a function into a series of sine functions, where each sine function is multiplied by a coefficient and either summed or integrated. The Discrete Fourier Transform (DFT) was created as a way to numerically approximate the Fourier transform. The FFT algorithm is a highly effective means of implementing the DFT. To distinguish between even and odd numbered components during the FFT process, a technique called bit reversal is utilized. This involves reversing the order of binary bits in the decimal digits. The FFT is a powerful mathematical tool used to break down a raw audio signal into its constituent frequencies. By using the FFT, we can identify the fundamental frequencies and pitches present in the signal. Our objective is to determine the frequencies of the individual sinusoids that make up the musical tone, as this is directly related to how we perceive its pitch. To ensure the accuracy of the FFT, it must operate over a long enough period of time (T) to distinguish between lower pitches of an instrument. The resolution of the frequency is determined by the inverse of T, meaning that enough time must pass to complete one full cycle of a frequency in order to accurately resolve that frequency.

II. Methodology

Figure 2 presents a flow chart that provides a summary of the methodology used in this research project. The process of using the FFT algorithm to extract notes from "Fur Elise" song involves several steps. First, a small window is taken from the audio signal. To enhance the signal-to-noise ratio and reduce computational costs, the audio signal is downsampled before the FFT transformation, and a threshold value is utilized to differentiate between the notes and background noise in the signal. The next step is detecting the notes, which involves padding the audio signal with zeros to ensure accurate frequency estimation. The FFT algorithm is then applied to transform the audio signal from the time domain to the frequency domain representation, enabling the analysis of its frequency content. Once the audio signal is transformed, the individual notes can be identified by analyzing the frequencies of the different components of the signal. This involves detecting the frequency peaks in the frequency domain representation of the signal, which corresponds to the fundamental frequency and its harmonics. By matching these peaks with a set of predefined piano note frequencies, the frequencies of the individual notes can be determined. Finally, to recreate the song, each captured piano note is played back for a duration of 0.5 seconds by generating a new audio signal with the corresponding frequencies and durations for each note. Overall, the process of extracting notes from music using the FFT algorithm entails various steps, such as downsampling, averaging, thresholding, padding, frequency analysis, identification of piano notes, and note recreation. The approach for carrying out this task can be broken down into the subsequent stages:

Sampling: When you speak or play a musical instrument, the sound wave produced is an analog wave that fluctuates with air pressure. To save this sound wave onto a computer, it needs to be recorded at fixed time intervals with discrete values. The act of recording time values in a discontinuous manner is known as sampling, whereas the act of recording pressure values discontinuously is called quantizing. Recording studios commonly use a sampling frequency of 48 kHz, while CDs use 44.1 kHz. To achieve accurate reproduction, signals should be sampled at a rate that is twice the highest frequency present in the signal. Since the human ear can hear frequencies ranging from 20-20,000 Hz, typical sampling rates fall within the range of 40 kHz.

DownSampling: The terms downsampling and compression are commonly used in digital signal processing to refer to the process of resampling in a multi-rate digital signal processing system. Downsampling is the process of reducing the bandwidth (filtering) and sample rate of a sequence of signal samples or a continuous function. By doing so, it generates an approximation of the sequence that could have been obtained by sampling the signal at a lower rate or density. It is similar to reducing the resolution of a photograph. The term "decimation" originally referred to removing every tenth one, but in signal processing, it now means keeping only every tenth sample. This factor is utilized to either

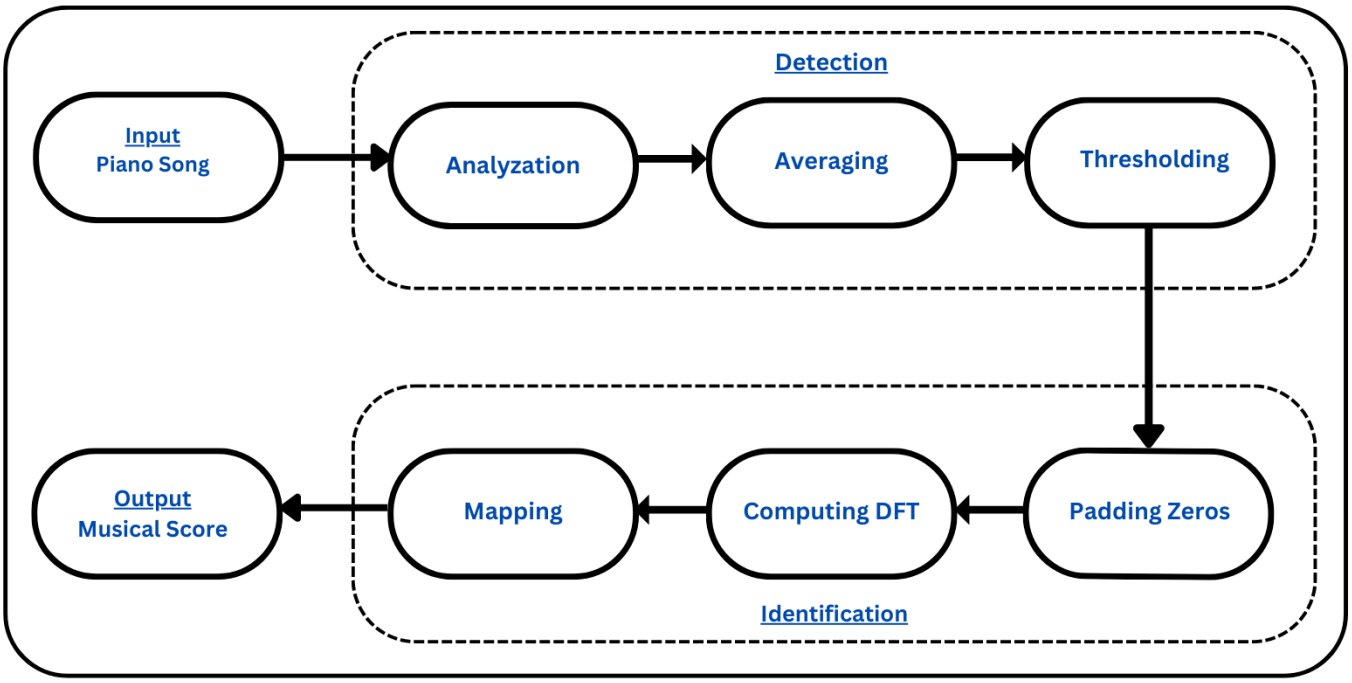

Fig. 2. A visual representation in the form of a flowchart that presents a summary of the methodology

multiply the sampling interval or divide the sampling rate. For instance, if the sampling rate of compact disc audio is 44,100 samples per second and it is decimated by a factor of 5/4, the resulting sample rate becomes 35,280. A decimator is a system component that carries out the process of decimation. Additionally, compression, which is also known as decimation by an integer factor, refers to this process.

Averaging: To deal with the large number of samples and fluctuations present in a song, the first step is to perform averaging. This requires computing the mean value for every 100 samples and setting it as the value for the first sample. The same process is then repeated for every subsequent 100 samples. This method helps to decrease the sample count and remove any abrupt variations in the signal. Nonetheless, if the signal's decay rate is gradual, the averaged signal will be more compact. Conversely, if the signal's decay rate is rapid, the averaged signal will indicate the envelope of the initial signal.

Thresholding: There are two techniques that can be employed to detect peaks from the averaged signal, namely constant thresholding and adaptive thresholding. Constant thresholding involves setting a single value that can detect the maximum number of peaks in the signal. However, using a fixed threshold value may not be optimal for all notes, as some notes may have values that are too low or too high, which can result in merged peaks or missed peaks. In order to solve this problem effectively, a technique called adaptive thresholding was created. This method involves modifying the threshold value according to the specific attributes of the signal in the surrounding area, in order to achieve precise identification of peaks.

Padding with Zeroes: To obtain accurate results, it is necessary to pad the original signal with zeros before computing its discrete Fourier transform (DFT). Simply cropping the signal and applying the DFT do not yield accurate results. By padding the signal with zeros before and/or after the cropped portion, different results can be obtained depending on the placement and length of the zero-padded segments. After experimenting with different variations, it was found that the closest results were obtained when the same length of zeros was added to both ends of the trimmed segment. The musical notation is then determined by utilizing the DFT of the signal that is obtained.

Computing DFT: By using a Fourier transform, we can break down a complex signal, like a musical tone, into individual sinusoids. This technique involves performing several integrals and analyzing a continuous signal. In order to perform a Fourier transform on a signal that has been sampled, we need to use the DFT instead of the continuous Fourier transform. The FFT is the most widely used method for implementing the DFT. While the DFT has a runtime of O(N^2), the FFT produces the same output as the DFT but with a much faster execution time. The FFT has a computational complexity of O(NlogN), which is faster than the DFT's O(N^2) runtime. To discuss the FFT accurately, it is important to establish precise definitions of its input parameters. The input to an FFT involves the following factors: Sampling frequency (fs): The number of samples captured per second when analog data is

converted to digital data. It is measured in Hz, and sometimes simply referred to as ”fs.” Number of data points (n): The quantity of data points in the input. Total time (T): The duration of the sample being taken. It is measured in seconds and can be calculated by dividing n by fs. Frequency resolution (f): The precision with which frequencies can be determined. It is defined as the inverse of T. The objective is to identify the frequencies of the individual sine waves that form the musical tone since this is related to the perception of its pitch. To make the FFT effective, it needs to work for a duration (T) that is long enough to distinguish the lower pitches of an instrument. The frequency resolution is the reciprocal of T, as enough time is required to complete one full cycle at a frequency in order to precisely detect its frequency.

Identification: The previous steps involved identifying the moments at which the notes were played by determining the time intervals between them. The next step is to determine the specific notes being played during these intervals.

III. Results

The input song ”FurElise” was analyzed and 29 frequencies were detected in the sequence of 313.8Hz, 296.7Hz, 313.8Hz, 296.7Hz, 313.8Hz, 234.3Hz, 279.6Hz, 248.4Hz, 209.2Hz, 104.6Hz, 124.7Hz, 156.9Hz, 209.2Hz, 234.3Hz, 156.9Hz, 197.1Hz, 234.3Hz, 248.4Hz, 156.9Hz, 313.8Hz, 295.7Hz, 313.8Hz, 296.7Hz, 313.8Hz, 234.3Hz, 279.6Hz, 248.4Hz, 209.2Hz, and 104.6Hz. The graphs of the output frequencies of the FurElise song at the beginning and end are shown in Figure 3 and Figure 4. The corresponding piano notes for these frequencies were identified and stored in a variable, namely Eb4, D4, Eb4, D4, Eb4, Bb3, Db4, B3, Ab3, Ab2, B2, Eb3, Ab3, Bb3, Eb3, G3, Bb3, B3, Eb3, Eb4, D4, Eb4, D4, Eb4, Bb3, Db4, B3, Ab3, and Ab2. The extracted piano notes were played for a duration of 0.5 seconds each to recreate the input song, and the recreated song sounds almost identical to the original input song.

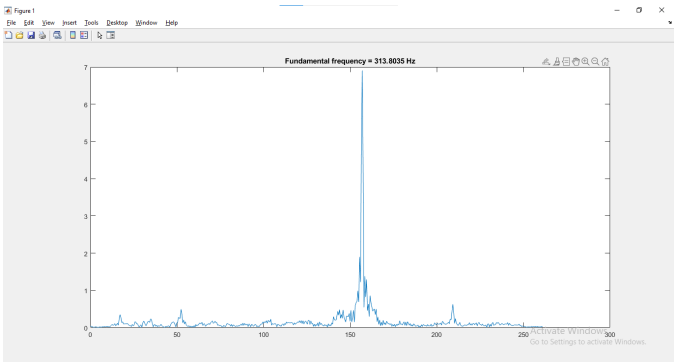

Fig. 3. First frequency of FurElise song

IV. Conclusion

This article describes the successful development and implementation of a user-friendly piano note recognition system using FFT. The intended users of the system include both music enthusiasts and professional musicians who need to identify notes quickly. The system has demonstrated high accuracy in hit detection for moderately-paced songs up to eighth notes, and high success rates in identifying single notes and determining their tempo. However, the project has some limitations such as poor note-hit detection in fast and busy songs, overlapping note-hits causing single notes to appear as multiple shorter notes, and anomalies caused by instrument faults and non-ideal behavior. Additionally, the system is not real-time and takes around 60 seconds to calculate and recreate output music using a MATLAB-based code on a 2.6 GHz computer with 16GB of RAM. Therefore, improvements in efficiency may be necessary to make the system more suitable for real-time applications.

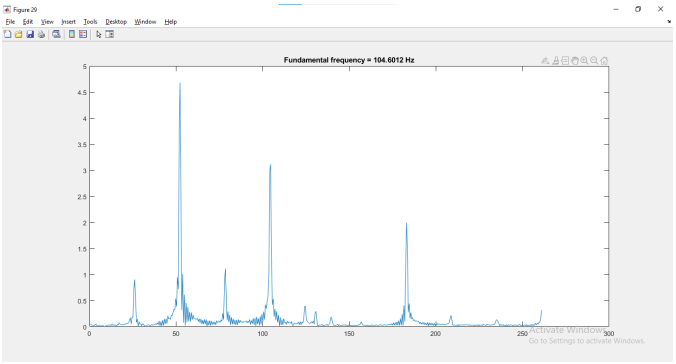

Fig. 4. Last frequency of FurElise song

V. Appendices

Access to files required for this Digital Signal Processing project is available through this link (https://drive.google.com/drive/folders/1D9ZYeNzoJnQVUeAVXNLFyNo_wxAvjiG0?usp=share_link).

References

[1] https://brightstarmusical.com/the-frequencies-of-the-musical-scale/

[2] Leo Prasanth L., Uma E., ”A FRAMEWORK FOR PIANO NOTES ANALYSIS USING DIGITAL SIGNAL PROCESSING”, International Journal of Creative Research Thoughts (IJCRT), Volume 11, Issue 2 February 2023, ISSN: 2320-2882, https://ijcrt.org/papers/IJCRT2302528.pdf

[3] Jay K. Patel, E.S. Gopi, ”Musical Notes Identification using Digital Signal Processing”, Procedia Computer Science, Volume 57, 2015, Pages 876-884, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2015.07.499

[4] Say Wei Foo, Wei Thai Lee, ”Recognition of piano notes with the aid of FRM filters,” First International Symposium on Control, Communications and Signal Processing, 2004., Hammamet, Tunisia, 2004, pp. 409-413, doi: 10.1109/ISCCSP.2004.1296315

[5] E.N.V PURNA CHANDRA RAO, I. NARASIMHA RAO, ”MUSIC NOTE RECOGNITION USING FFT”, CMR College Of Engineering & Technology, Kandlakoya, Medchal Road, Hyderabad-501401, https://www.scribd.com/document/602796520/Music-Note-Recognition-Using-Fft

[6] Wikaria Gazali, Djunaidy Santoso, (2011). ”Designing Application Programs Sheet Music Using Fast Fourier Transform Method”. Lecture Notes in Engineering and Computer Science. 2189

[7] https://en.wikipedia.org/wiki/Piano_key_frequencies